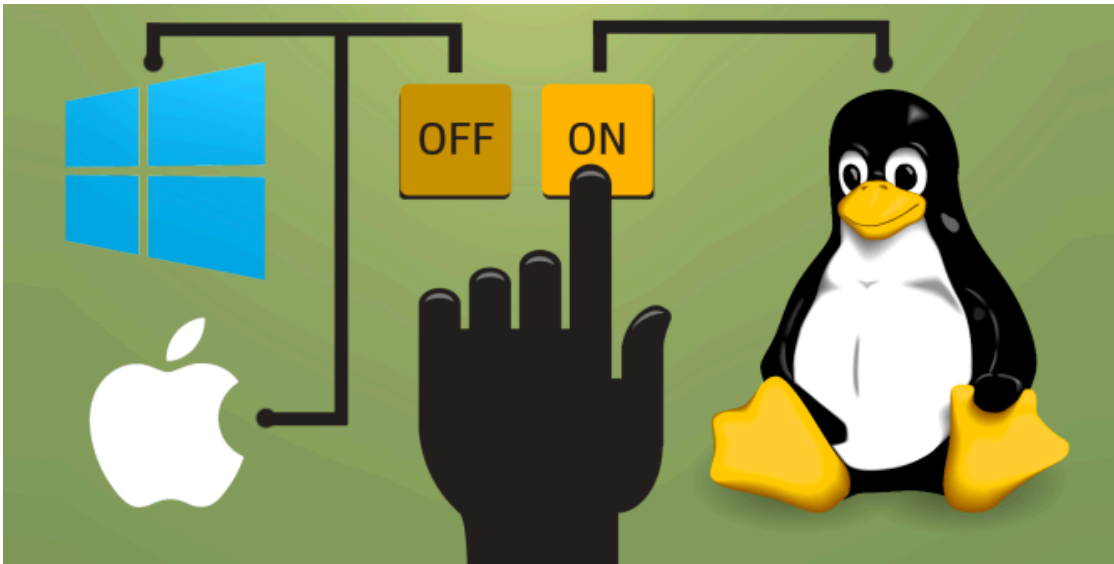


# An introduction to Linux

<https://forge.uclouvain.be/elic/learning.git>



**Pierre-Yves Barriat**

ACELI Training Sessions April 10th, 2025

# What is UNIX ?

- Operating System
  - | Windows 10 (Microsoft), MacOSX (Apple), Android (Google), etc
- UNIX is a (family of) Operating System
- Invented by AT&T Bell Labs in late 60's
- Currently there are different versions and variants of UNIX
  - | Solaris, AIX, HP-UX, etc.
- UNIX is **not** free or Open Source: "GNU is Not UNIX"

# What is Linux ?

- GNU (80's) is a free, open source version of the UNIX OS, but without the most important element: the **kernel**
- Linux kernel was developed in 1991 by Linus Torvalds, a Finnish graduate student
- The association GNU/Linux is an operating system (say just "Linux") and provides an alternative to commercial operating systems
- Linux exists without GNU (eg Android) : used to power a multitude of systems... from your phone to your smart fridge

- **Core Components**

The "Linux Kernel" is the core of the operating system, which interacts directly with the hardware.

Others kernels before Linux: Minix, GNU Hurd, BSD, etc

- **GNU Tools**

Linux is often used with tools from the GNU project (like compilers, shell, and libraries), making it a complete operating system when combined with the kernel.

- **Open Source**

The source code for Linux is freely available, meaning you can modify and distribute it.

# What is a Linux distribution ?

- Many versions of Linux
  - Red Hat, Debian, Suse, etc
- But one common linux kernel: **kernel** is like an **engine**. A distribution is an actual car that hosts the engine
- Distributions differ from
  - the application/management layer
  - GUI (Graphical User Interface = desktop environment)
  - software packages
  - help-desk, support, language

# Popular Linux Distributions

- **Ubuntu:** User-friendly, ideal for beginners.
- **Fedora:** Cutting-edge features and software.
- **Debian:** Stable and reliable, popular in scientific environments.
- **Rocky:** Enterprise-focused, ideal for servers.
- **Arch Linux:** Lightweight and customizable (for advanced users).

## Why Multiple Distros ?

Different distributions cater to different needs: some are easier for beginners, while others are tailored to performance, security, or customization.

# Why Linux for scientists ?

As a scientist, your workflow involves complex simulations, data analysis, and working with scientific tools.

Linux provides a stable, powerful, and open environment tailored for research purposes.

Here's why it's better than Windows for scientific work.

- **Performance and Efficiency**

Linux is known for being fast and efficient. It handles large computational tasks better than Windows, making it ideal for simulations and data-heavy research.

- **Flexibility**

You can customize Linux to meet the specific needs of your research, from installing specific software to adjusting the environment.

- **Better Use of System Resources**

Unlike Windows, Linux runs with minimal overhead, giving you more resources for your tasks.



# Why Linux for all ?

- Linux is **free** and open-source
- Linux is supported on older computers (perf & updates)
- Linux has many more free applications

Security : there are very few viruses for Linux

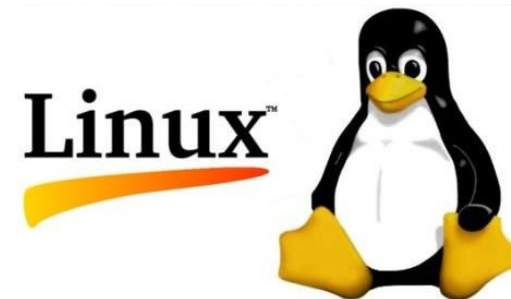
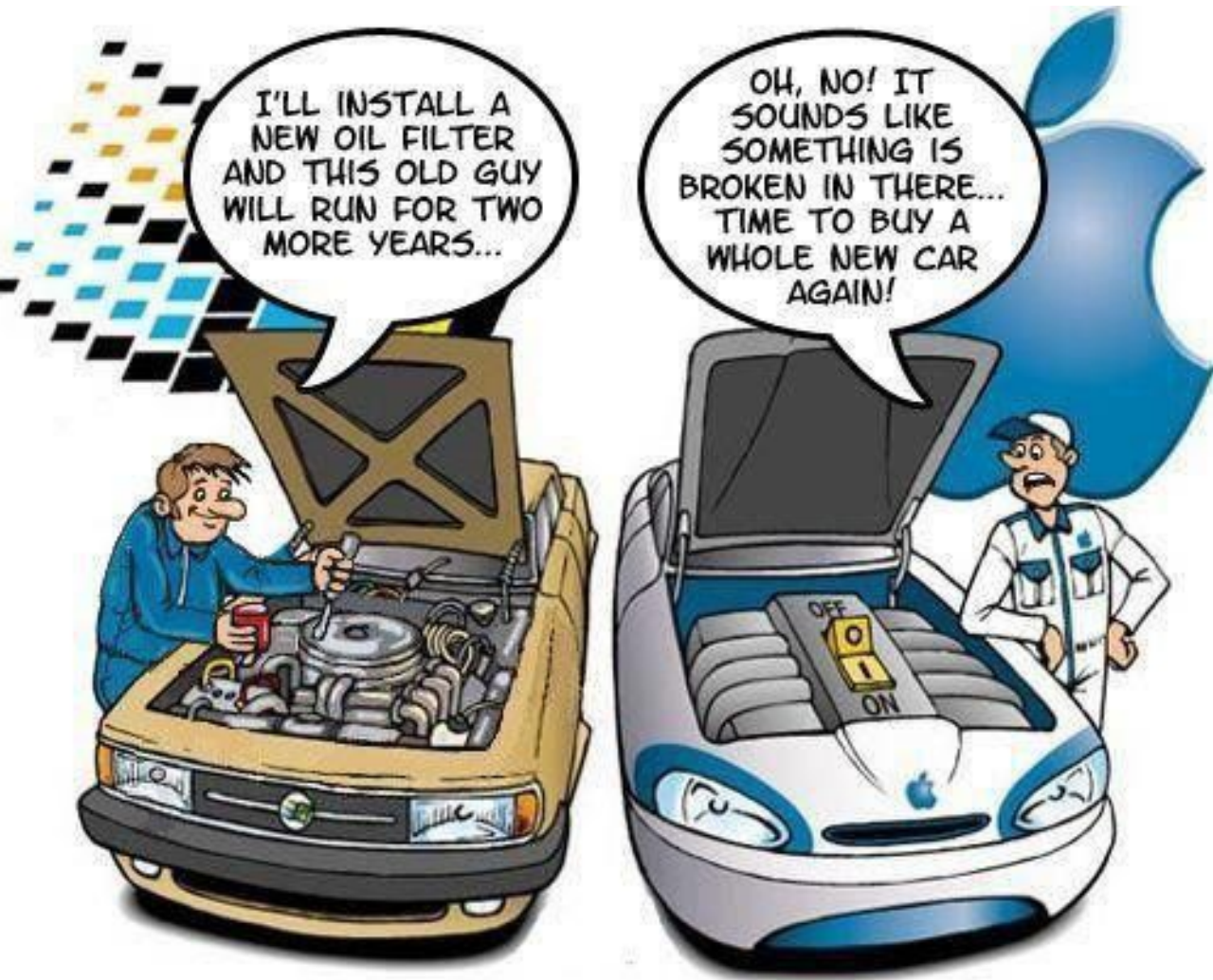
Privacy : most Linux distributions don't collect your data at all

Reliability : if you want to stop something, you really can

Updates : package manager

Customisation : you can make Linux look, feel and behave as you want it to

Command Line Interface: faster & efficient



# Linux Statistics

- 47% of professional **developers** use Linux-based operating systems (Statista)
- Linux powers 39.2% of **websites** whose operating system is known (W3Techs)
- Linux powers 85% of **smartphones** (Hayden James)



## Primary Operating Systems Among Professional Developers



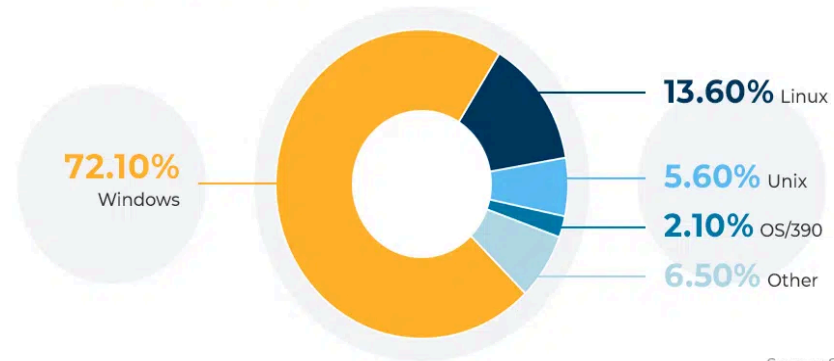
Source: Statista

## Leading Desktop Operating Systems Worldwide by Market Share



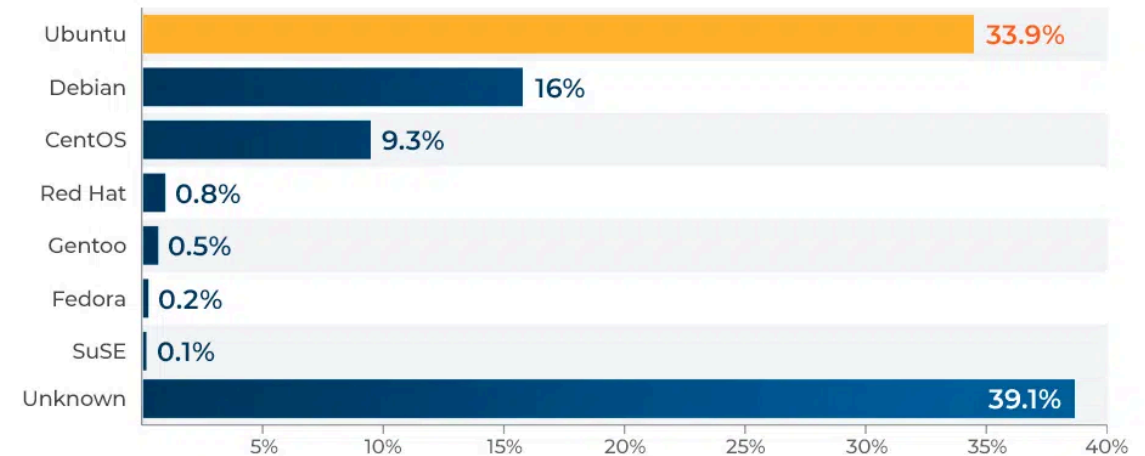
Source: Statista

## Server Market Share Worldwide by Operating System



Source: Statista

## Top Linux Subcategories by Market Share



Source: W3Techs

# Robust software support

- **Scientific Software**

Many popular scientific tools (e.g., ROOT, MATLAB, Python, R, and SciPy) work seamlessly on Linux.

- **Package Management**

Linux distributions offer package managers like APT, YUM, and Pacman, making it easy to install, update, and manage software.

- **Compilers and Libraries**

Linux has native support for most programming languages and compilers (e.g., GCC for C/C++, Fortran) and scientific libraries.



# Better for High-Performance Computing (HPC)

- **Parallel Computing**

Linux is optimized for high-performance computing, supporting parallel processing and multi-core usage that is crucial for simulations and data analysis.

- **Supercomputing**

The majority of the world's supercomputers run Linux, making it the go-to operating system for large-scale simulations and scientific research.

- **Cluster Management**

Tools like OpenMPI and SLURM work flawlessly on Linux for managing distributed computing environments, common in physics research.

# Why not Linux ?

1. Hardware compatibility (printers, etc)
2. Missing famous software (MS Office, Adobe, CAO, etc)
3. Gaming

## Workarounds

1. Many devices "Linux compatible"
2. Emulation (eg virtualbox), online usage, alternatives (GIMP)
3. More and more games on Linux (Steam OS) and emulation

# Distribution: why Ubuntu ?

- easy to install and easy to use
- easy to maintain and update
- useful applications
- looks nice
- wide variety of supported applications
- strong community support
- better driver support
- LTS and staging releases available as per user needs

Get Ubuntu LTS : [ubuntu-22.04.5-desktop-amd64.iso](#)



# Test Linux inside Windows: how ?

Using a virtual machine: test Linux without changing anything to your computer

You need to install a VM and then install the Linux inside

Take a look in the `learning-vscode` folder in:

<https://forge.uclouvain.be/elic/learning.git>

Using WSL in Windows 11 in order to use Linux

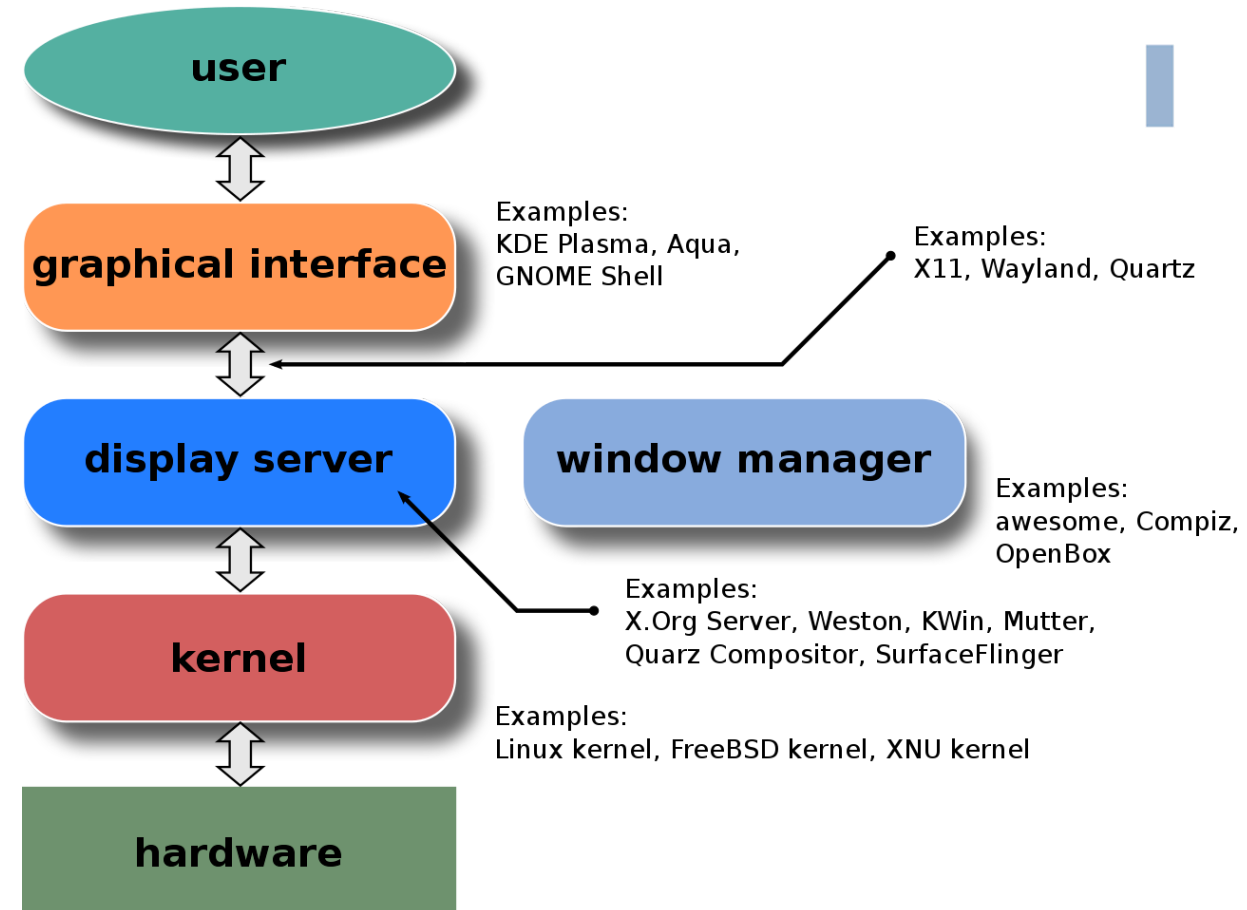
# Linux GUI

Most Linux distributions come with a desktop environment (e.g., GNOME, KDE) that provides a graphical interface similar to Windows.

You can interact with files, applications, and settings using windows, icons, and menus.

A desktop environment contains a **display server**, a **window manager** (manage windows, panel, menus, dash interfaces and core applications), and a **desktop environment** (eg status bars, drag-and-drop capabilities, etc)

These 3 items are **bundled together** to make what is known as a **GUI environment**



# Using Linux

**Using Linux through a GUI environment is similar to Windows or MacOSX**

Linux users do not install software the same way that Windows users do:

Linux has a tool known as a `package manager`

A package manager provides a way to search for software, install software, keep the software up to date and remove the software

Similar to Google or Apple store

# Linux CLI

Using Linux with **GUI**, it's already great !

Using **CLI** = **unlock the power of Linux** !

**CLUI** = **C**ommand **L**ine **U**ser **I**nterface (= **CLI**)

- It is one of the many strengths of Linux and can be more efficient than using the GUI
- The CLI allows you to interact with the system by typing text-based commands into a terminal.
- While more intimidating at first, the CLI is extremely powerful for advanced operations, automation, and performance.

## Which to Use ?

GUI is great for beginners or tasks that require minimal technical knowledge.  
CLI is essential for advanced users, system administration, and scripting.

- CLI allows users to be independent of distros (or UNIX systems like OSX)
- CLI saves system resources which are consumed by GUIs
- CLI allows users to easily work at distance (SSH)
- CLI allows developers to join together simple (and less simple) commands to do complex things and automate... whatever you want to

People tend to think command line is difficult. It's not.

# Linux Shell

A **shell** is a program that takes commands from the keyboard and gives them to the operating system to perform

The main function is to interpret your commands = **language**

The **bash** shell is one of several shells available for Linux

Learning the shell:

*"When you are a child you use a computer by looking at the pictures. When you grow up, you learn to read and write"*

It's more or less like SMSing to your PC, telling it what to do

# Linux Shell

Shells have some built-in commands

A shell also supports programming constructs, allowing complex commands to be built from smaller parts = **scripts**

Scripts can be saved as files to become new commands

| many commands on a typical Linux system are scripts

An open terminal show you a **PROMPT** waiting for your commands

Commands can have **options** and parameters

All your commands are saved in a **history**



# Linux Shell Demo

Rename many files at once ?

```
mmv '*.JPG' '#1.jpg'
```

Download a youtube video ?

```
yt-dl https://www.youtube.com/watch?v=G7KNmW9a75Y&ab_channel=MileyCyrusVEVO
```

Convert color pictures in BW at once ?

```
#!/bin/bash
for file in *.jpg
do
    convert ${file} -colorspace Gray "${file%.*}_bw.jpg"
    echo "${file}... converted"
done
```

# Online Linux Demo

- Run Linux or other Operating Systems in your browser
- Run bash (and others) scripts online
- Free online containers and virtual machines that run full Linux systems
- Free GNU/Linux Online Terminal learning platform
- CISM/ELIC Jupyter portal

you need a UCLouvain/CISM account: see [here](#)

# What you'll learn

- Navigating the File System
  - | Get up and running with the CLI by navigating directories and files
- Viewing and Changing Files and Directories
  - | Learn to manipulate directories and files from the CLI
- Configuring the Environment
  - | Learn to configure the environment using the CLI
- Accessing Linux remotely
  - | Learn to use SSH (basics)

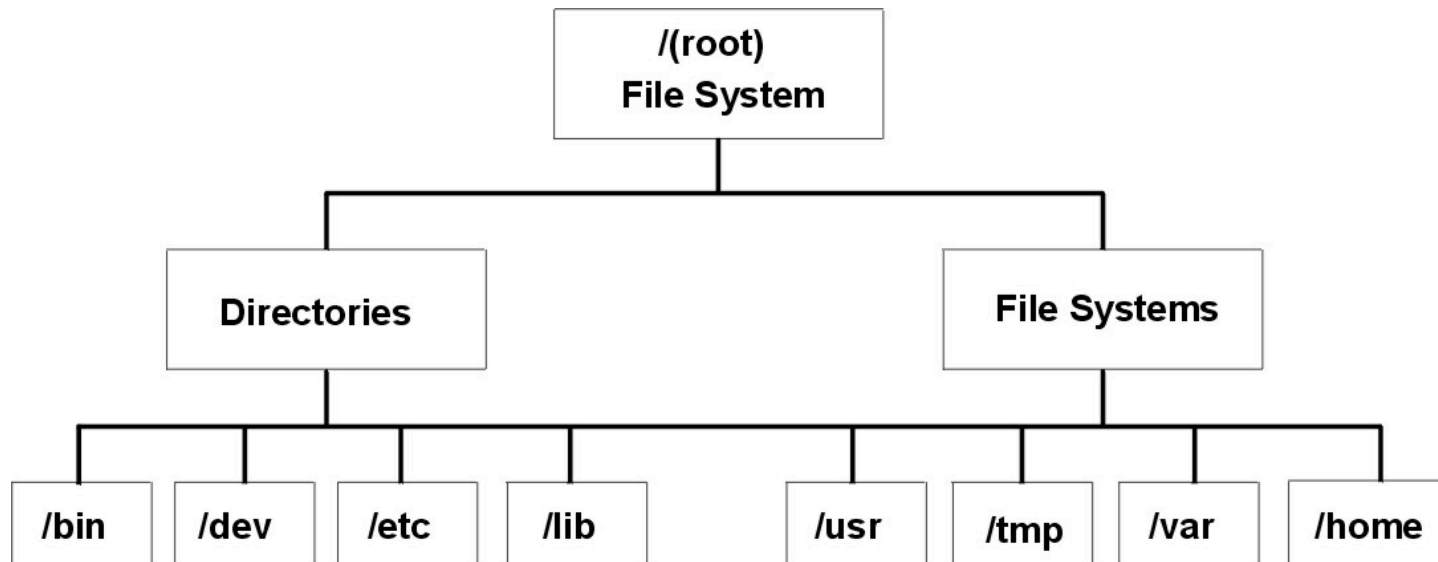
# Navigating the File System

A file system is a section of hard disk that has been allocated to contain files

it's arranged like hierarchical tree structure

Files are grouped in the directory structure

The top of the tree is called **root** and **/** is used to present the **root**



## Absolute paths

In the tree `/users/usern/file1` is an absolute pathname

## Relative paths

If you are already in the `users` directory, the relative pathname for `file1` is `usern/file1`

- `~` (tilda) points to the user's **home directory**. Useful if you are logging into a workstation with many users. It's the **default working directory** when you log in. If you are user `usern`, then `/users/usern/file1` is the same as `~/file1`
- `.` refers to the current directory
- `..` refers to the parent directory

# Basic commands

- `ls` : lists folders/files in a directory
- `cd` : change directory
  - use `cd name` to navigate to directory name
- `pwd` : print working directory. Prints the path of the current directory
- `du` : disk usage. Shows the disk usage of the current directory
- `man` : manual
  - use `man name` to bring up a manual entry for command or program called `name`

## Creation

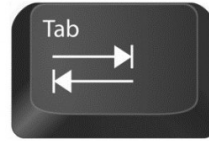
- `mkdir` : use `mkdir name` to create a new directory named `name` in the current directory
- `cp` : use `cp file1 file2` to create `file2` which is a copy of `file1`  
| can also use `cp -r` to copy whole directories
- `mv` : move = same as copy, but deletes the original file

## Deletion

- `rm` : delete files ( cannot recover your files after removed them ! )  
| can also use `rm -rf` to remove whole directories

**Be careful : there is no trash in CLUI**

- `tab` is used for auto-complete



If a file/directory name was partly typed in, **tab will auto-complete** it

If there are multiple options, tab will auto-complete up to the point where the options branch and show you a list of possible options

- `*` is used as a wild card

`rm blah*` removes all files which start with `blah` : eg `blah1` , `blah2` , and `blahblah` would all be removed

using `cp public/* private/` copies all files in a `public` directory into a `private` directory, and keeps all file names intact



# File permissions

## Groups

Each file and directory has three user based permission groups

- **owner** : the `owner` permissions apply only the owner of the file or directory, they will not impact the actions of other users
- **group** : the `group` permissions apply only to the group that has been assigned to the file or directory, they will not effect the actions of other users
- **all users** : the `all users` permissions apply to all other users on the system, this is the permission group that you want to watch the most

## Types

Each file or directory has three basic permission types

- **read** : the `read` permission refers to a user's capability to read the contents of the file
- **write** : the `write` permission refer to a user's capability to write or modify a file or directory
- **execute** : the `execute` permission affects a user's capability to execute a file or view the contents of a directory

The following command :

```
ls -l
```

gives :

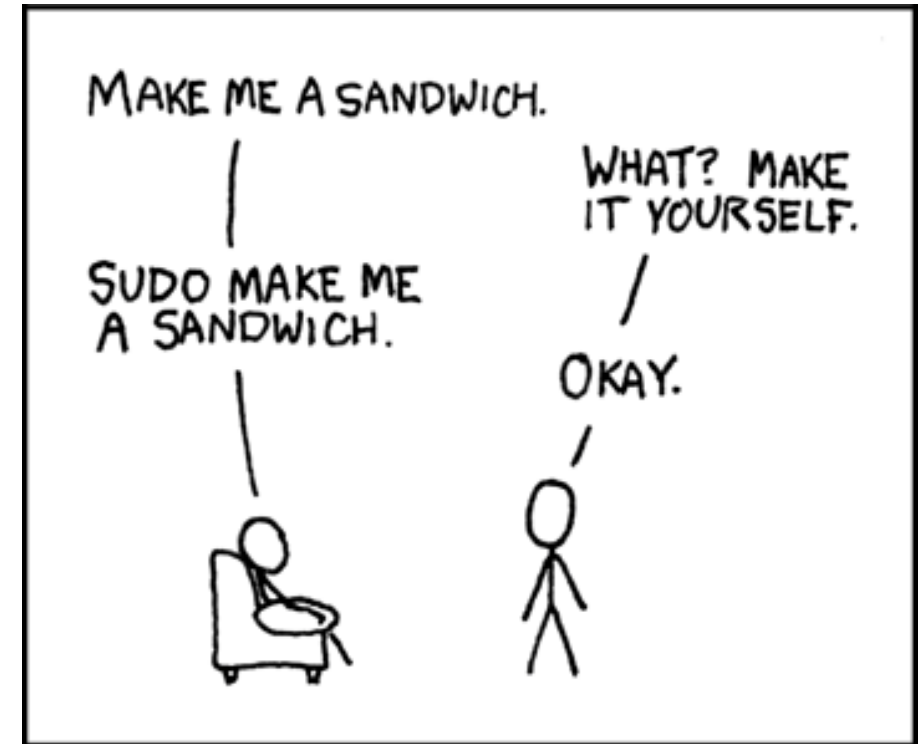
1	2	3	4	5	6	7
-rwxr-xr-x	1	dupont	grpelic	3528	2022-08-04	file_1
drwxr-xr-x	3	dupont	grpelic	512	2022-01-02	dir_1
lrwxr-xr-x	2	dupont	grpelic	210	2020-12-16	short -> /data
-rw-----	1	dupont	grpelic	4587	2022-12-04	file_2

# Linux privileges

Linux has a robust permissions system: this is a very good thing, as it enables a **clear separation** of roles among users

Especially between the `root` user and your `standard` user

Sometimes, though, you might want your standard user to have **some** or **all** of root's privileges : this is accomplished with `sudo`



# Shell syntax rules

Shells use three **"standard I/O streams"**

- `stdin` is the standard input stream, which provides input to commands.
- `stdout` is the standard output stream, which displays output from commands.
- `stderr` is the standard error stream, which displays error output from commands.

Shell has several **meta-characters** and **control operators**

| `|`, `&`, `>`, `;`, etc.

# Environment

In a bash shell many things constitute your environment

- the form of your prompt
- your home directory and your working directory
- the name of your shell
- functions that you have defined
- etc.

Environment includes many variables that may have been set **by bash** or **by you**

Access the value of a variable by prefixing its name with `$`

# Environment variables

- `USER` : the name of the logged-in user
- `UID` : the numeric user id of the logged-in user
- `HOME` : the user's home directory (similar to `~` )
- `PWD` : the current working directory
- `SHELL` : the name of the shell

Set a shell variable : typing a name followed immediately by an equal sign ( `=` )

| if the variable exists, you will modify it to assign the new value

You can use special files to control bash variables : `$HOME/.bashrc`

# Remote Linux Access

SSH (or **Secure SHell**) is a **protocol** used to securely log onto remote systems

the most common way to access remote Linux and Unix-like servers

VNC (or **Virtual Network Computing**) is a **software** that allows a personal computer's desktop environment to be run

Aside from bandwidth, latency and security issues (which can vary a bit), the big differences are the functionality

- VNC exports a whole session, desktop and all (GUI)
- SSH runs a single program (CLUI) and show its windows on your machine



# Remote Linux using SSH

You need:

- an access to the distant machine : login/password  
or a login with SSH keys (with passphrase)
- the hostname or the IP address of the distant machine
- and (of course) a UNIX terminal

```
ssh -X pbarriat@coriolis.elic.ucl.ac.be
```

```
ssh -X -i ~/.ssh/id_rsa.ceci pbarriat@gwcism.cism.ucl.ac.be
```

# Remote Linux from Windows

Using a SSH client to reach a distant Linux Workstation

Take a look in the `learning-vscode` folder in:

<https://forge.uclouvain.be/elic/learning.git>

see "VS Code nice extensions"

`Remote - SSH` : lets you use any remote machine with a SSH server

# Linux text editors

## GUI

- nedit (simple text editor available in most distributions)
- Kate, Gedit (KDE, Gnome)

## CLI

- vi (available in all Unix systems) and **vim** (vi improved)
  - Difficult to learn for beginners used to graphical text editors
  - Very productive for power users
  - [Vim Cheatsheet](#)
- nano (friendly and easier to learn)

# Scripting

How to do a backup ?

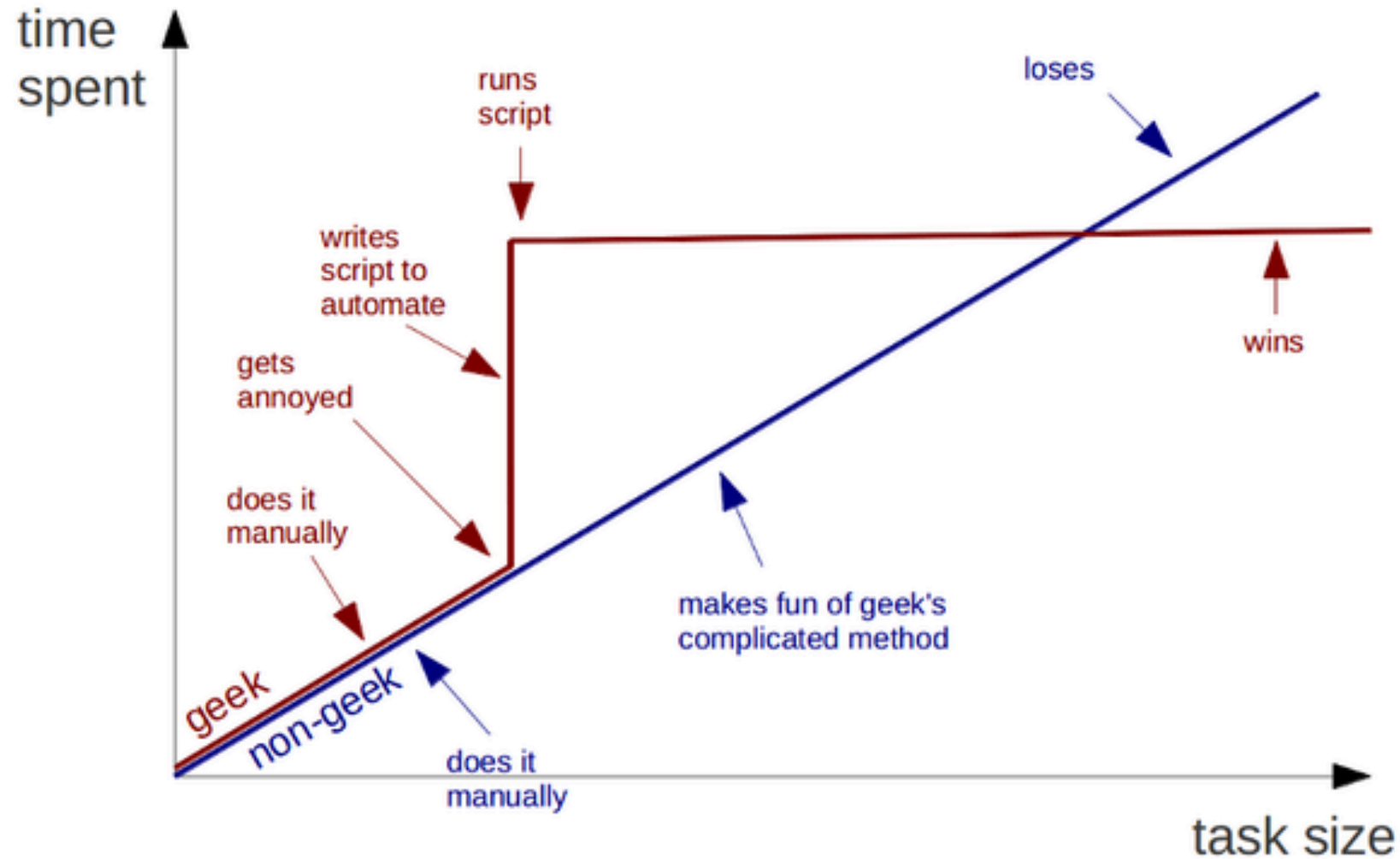
- with Dropbox or Google Drive ?
- with a private cloud such as Nextcloud ?
- with an other drive and/or an external drive ?

Backup on external drive ?

- manually ?
- with Windows tools ?
- with external softwares ?

**Why don't do that with a simple linux script ?**

## Geeks and repetitive tasks



# Conclusions

Advantages of Linux	Disadvantages of Linux
Cost	Not easy to master ( <b>CLI only</b> )
Security and robustness	Hardware compatibility issues ( sometimes )
Freedom	Not compatible with some Windows software
Software	
Development	

## Need to know more about available Linux applications ?

Check out the list of the best Linux software

## Need help with bash scripting ?

- Bash Scripting Tutorial for Beginners
- My favorite CLUI cheatsheet
- Bash scripting cheatsheet

## Need help with Ubuntu ?

The massive community is one of Ubuntu's biggest strengths

Visit <https://askubuntu.com/> or <https://answers.launchpad.net/>